(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 102 50 179 A1** 2004.05.13

(12) # Offenlegungsschrift

(21) Aktenzeichen: **102 50 179.3**
(22) Anmeldetag: **28.10.2002**
(43) Offenlegungstag: **13.05.2004**

(51) Int Cl.[7]: **G06F 13/00**
G06F 9/06

(71) Anmelder:
Océ Printing Systems GmbH, 85586 Poing, DE

(74) Vertreter:
Schaumburg und Kollegen, 81679 München

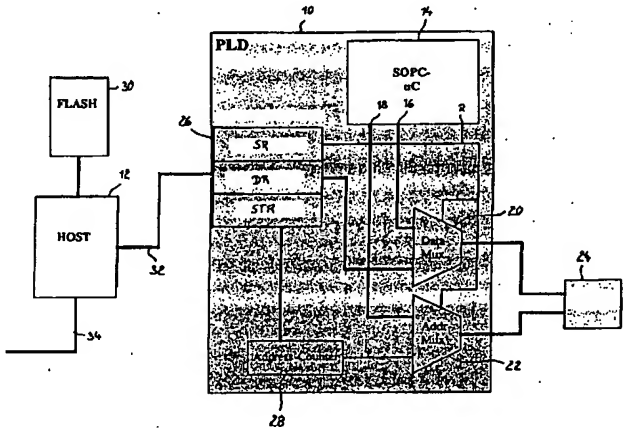
(72) Erfinder:
Katterloher, Rainer, Dipl.-Ing., 84405 Dorfen, DE; Zimprich, Stefan, Dipl.-Ing. (FH), 85737 Ismaning, DE; Wallis, Christian, Dipl.-Ing. (FH), 80687 München, DE; Zollner, Werner, Dipl.-Ing. (FH), 85462 Eitting, DE; Sippel, Helmut, Dipl.-Ing. (FH), 81735 München, DE; Gräf, Andreas, Dipl.-Inform., 82194 Gröbenzell, DE

Prüfungsantrag gemäß § 44 PatG ist gestellt.

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

(54) Bezeichnung: **Verfahren und Einrichtung zum Betreiben eines Datenverarbeitungs-Systems bei sich ändernder Firmware**

(57) Zusammenfassung: Beschrieben wird eine Einrichtung zum Betreiben eines PLD-Bausteins (10), der einen Mikrocontroller (14) umfaßt, der auf einen RAM-Baustein (24) zugreift. Ein Host-Prozessor (12) greift während einer Ladephase auf den RAM-Baustein (24) zu und lädt Programmdaten. In der Betriebsphase greift der Mikrocontroller (14) auf die Programmdaten im RAM-Baustein (24) zu.



Beschreibung

[0001] Die Erfindung betrifft Verfahren, Einrichtungen und Systeme zum Betreiben eines Datenverarbeitungssystems, wobei die Firmware unterschiedliche Versionen haben kann.

[0002] In Datenverarbeitungs-Systemen, die beispielsweise bei digitalen Druckern und Kopieren eingesetzt werden, sind die Geräteeinheiten in eine Vielzahl von Baugruppen oder Modulen unterteilt, die jeweils durch Software-Module gesteuert und verwaltet wird. Diese Software-Module enthalten häufig eine Firmware, die in Folge von Weiterentwicklungen oder anderweitigen Änderungen geändert und ausgetauscht werden. Hierbei entsteht das Problem, dass die Firmware auf einfache Weise ladbar sein muss und die verschiedenen Versionen der Firmware so gestaltet sein müssen, dass die verschiedenen Software-Module ordnungsgemäß miteinander kommunizieren und zusammenwirken können.

[0003] Aus der US-2002/0067504 A1 ist ein Verfahren und eine Einrichtung zum automatischen Upgraden eines Treiberbausteins für Drucker bekannt. Zunächst wird die aktuell in einer Arbeitsstation vorhandene Version eines Druckertreibers identifiziert, danach wird eine entfernte Station kontaktiert, in der nach neueren Versionen des Druckertreibers gesucht wird. Falls eine solche neuere Version vorhanden ist, wird die neuere Version des Druckertreibers über ein Datennetzwerk geladen und als aktuelle Version des Druckertreibers in der Arbeitsstation gespeichert.

[0004] Es ist Aufgabe der Erfindung, Verfahren, Einrichtungen. und Systeme bereitzustellen, mit deren Hilfe auf schnelle Weise Firmware geladen werden kann. Ferner ist es Aufgabe der Erfindung, zu gewährleisten, dass geladene Software-Versionen mit weiteren Software-Bausteinen kompatibel sind.

[0005] Diese Aufgabe wird für eine Einrichtung durch den Anspruch 1 gelöst. Gemäß dieser Einrichtung wird ein PLD-Baustein, d.h. ein Programmable-Logic-Device, mit Hilfe eines Host-Prozessors angesteuert. Der PLD-Baustein enthält einen Microcontroller. In einer Ladephase lädt der Host-Prozessor Programmdaten zum Steuern des Microcontrollers in einen RAM-Baustein. In einer Betriebsphase greift der Microcontroller auf diese Programmdaten im RAM-Baustein ausgehend von einer ebenfalls übertragenen Startadresse zu. Auf diese Weise ist es nicht erforderlich, für den Microcontroller einen zusätzlichen ROM-Baustein für die Programmdaten vorzusehen. Weiterhin ist es auf einfache Weise möglich, sich ändernde Firmware schnell in den RAM-Baustein zu laden und damit dem PLD-Baustein neue Funktionen zuzuweisen.

[0006] Gemäß einem weiteren Aspekt der Erfindung wird ein Verfahren zum Betreiben eines PLD-Bausteins angegeben, mit der die bei der Einrichtung genannten Vorteile ebenfalls erreichbar sind.

[0007] Bei einem weiteren Aspekt der Erfindung wird ein Verfahren zum Betreiben eines Datenverarbeitungssystems angegeben. Das System umfasst eine zentrale Steuerung, die einen als Software-Modul ausgebildeten Versions-Manager enthält. Der Versions-Manager enthält in einem Verzeichnis Soll-Versionsdaten von verschiedenen Software-Modulen. Bei einer Änderung des Systems vergleicht der Versions-Manager Ist-Versionsdaten der verschiedenen Software-Module mit Soll-Versionsdaten. Bei einer Abweichung der Versionsdaten wird eine Fehlermeldung erzeugt. Auf diese Weise wird bei einem Ersatzteileaustausch oder einem Austausch der Firmware in den Softwaremodulen sichergestellt, dass die verschiedenen Softwaremodule und auch die Anwendungsumgebung aufeinander abgestimmt sind. Dies hat insbesondere Vorteile bei der Wartung und der Erneuerung der Datenverarbeitungs-Systeme für Drucker und Kopierer.

[0008] Gemäß einem weiteren Aspekt der Erfindung wird ein Verfahren zum Betreiben eines Datenverarbeitungssystems angegeben, bei dem das System mehrere Software-Module umfasst, die untereinander auf der Basis eines Netzwerk-Protokolls kommunizieren. Die Software-Module enthalten jeweils Software-Agenten, die Versionsdaten über das eigene Software-Modul und Soll-Versionsdaten über die für die Funktion des Systems benötigten weiteren Software-Module enthält. Es wird eine Versionsprüfung durchgeführt, bei der die Soll-Versionsdaten der benötigten Software-Module und die Ist-Versionsdaten dieser Software-Module auf Kompatibilität überprüft werden. Mit Hilfe dieser Kompatibilitätsprüfung wird sichergestellt, dass das System beim Zusammenwirken der verschiedenen Software-Module ordnungsgemäß funktioniert. Das beschriebene Verfahren ist dezentral angelegt, da die Daten in den Software-Agenten zu den verschiedenen Software-Modulen abgelegt und dort verfügbar sind.

[0009] Zum besseren Verständnis der vorliegenden Erfindung wird im folgenden auf die in den Zeichnungen dargestellten bevorzugten Ausführungsbeispiele Bezug genommen, die anhand spezifischer Terminologie beschrieben sind. Es sei jedoch darauf hingewiesen, dass der Schutzumfang der Erfindung dadurch nicht eingeschränkt werden soll, da derartige Veränderungen und weitere Modifizierungen an den gezeigten Vorrichtungen und/oder den Verfahren sowie derartige weitere Anwendungen der Erfindung, wie sie darin aufgezeigt sind, als übliches derzeitiges oder künftiges Fachwissen eines zuständigen Fachmannes angesehen werden. Die Figuren zeigen Ausführungsbeispiele der Erfindung, nämlich

[0010] Fig. 1 eine Anordnung mit einem PLD-Baustein, der von einem Host-Prozessor angesteuert wird, sowie einen einzigen RAM-Baustein;

[0011] Fig. 2 ein Druckersystem mit einer Vielzahl von Software-Modulen und einer zentralen Steuerung mit einem Versions-Manager; und

[0012] Fig. 3 eine Anordnung verschiedener Soft-

ware-Agenten mit Verweisen auf Versionsdaten der zugehörigen Software-Module.

[0013] Fig. 1 zeigt eine Anordnung mit einem PLD-Baustein 10, der von einem übergeordneten Host-Prozessor 12 angesteuert wird. Der PLD-Baustein 10 ist ein „programmable-logicdevice", welches eine Vielzahl programmierbarer Logikelemente enthält. Der PLD-Baustein 10 enthält einen SOPC-Baustein (SOPC: System on a programmable chip microcontroller) 14 als programmierbares Software-Modul mit einem Reset-Eingang R, Datenleitungen 16 und Adressleitungen 18. Diese Leitungen 16, 18 sind verbunden mit einem Daten-Multiplexer 20 bzw. einem Adress-Multiplexer 22, deren Ausgänge mit einem RAM-Baustein 24 verbunden sind. Dieser RAM-Baustein 24 dient einerseits als Aufbewahrungsort für die Programmdaten des SOPC-Bausteins 14 als auch als Arbeitsspeicher für diesen Baustein 14. Als Programmdaten sind Firmware-Daten verwendbar, die infolge unterschiedlicher Softwareversionen relativ häufig änderbar sind.

[0014] Weiterhin enthält der PLD-Baustein 10 eine Registerschnittstelle 26 mit einem Steuerregister SR, einem Datenregister DR und einem Startadressenregister STR. Das Startadressenregister STR greift auf einen Adresszähler 28 zu, der ausgangsseitig mit dem Adress-Multiplexer 22 verbunden ist. Das Datenregister DR ist mit dem Daten-Multiplexer 26 verbunden. Das Steuerregister SR ist mit dem Reset-Eingang R des SOP-Bausteins 14 sowie mit den Steuereingängen des Daten-Multiplexers 20 und des Adress-Multiplexers 22 verbunden.

[0015] Der Host-Prozessor 12 ist mit einem Flash-ROM-Speicher 30 verbunden. Dieser Flash-ROM-Speicher 30 kann in seinem Inhalt bei Bedarf geändert werden; typischerweise kann dieser Baustein rund 1000fach neu beschrieben werden. Der Flash-ROM-Baustein 30 enthält unter anderem die Programmdaten, die später den SOPC-Baustein steuern sollen. Der Host-Prozessor 12 ist über Datenleitungen 32 für Adressen, Daten und Steuersignale mit der Register-Schnittstelle 26 verbunden. Weiterhin hat der Host-Prozessor 12 Datenleitungen 34 zur Anbindung an ein Feldbus-System, über das er in ein vernetztes Prozessorsystem eingebunden ist und auf komfortable Weise nachgeladen werden kann.

[0016] Im folgenden wird die Funktion der in Fig. 1 beschriebenen Anordnung erläutert. Die Programmdaten für den SOPC-Baustein 14 liegen zunächst im Flash-ROM-Baustein 30 bereit und können von dort über denselben Ladealgorithmus nachgeladen werden, mit dem auch der Programmcode für den Host-Prozessor 12 nachgeladen wird. In einer Ladephase steuert der Host-Prozessor 12 über das Steuerregister SR den Reset-Eingang R des SOPC-Bausteins 14 an und hält diesen Baustein 14 in einem Reset-Zustand. Weiterhin wird der Daten-Multiplexer 20 so gechaltet, dass Daten vom Host-Prozessor 12 über die Leitung 32 und das Datenregister DR und den Daten-Multiplexer 20 in den RAM-Baustein 24 geladen werden, wobei die Adressleitung des RAM-Bausteins 24 über den Adress-Multiplexer 22 mit der Adressleitung des Adress-Zählers 28 verbunden ist. Dieser Adresszähler 28 wird über das Start-Adressenregister STR vom Host-Prozessor 12 geladen. Der Adresszähler 28 beginnt bei dieser Startadresse zu zählen und schaltet den Adress-Multiplexer 22 in seiner Adresse fortlaufend hoch. In dieser Ladephase wird der RAM-Baustein 24 mit Programmdaten für den SOPC-Baustein 14 geladen, wobei der Host-Prozessor 12 wortweise den im Baustein 30 stehenden Programmcode über die Registerschnittstelle 26 in das Datenregister DR einschreibt. Der Adresszähler 28 wird nach jedem Zugriff des Host-Prozessors 12 erhöht, nachdem die übertragenen Daten in den RAM-Baustein 24 übernommen wurden.

[0017] Bevor der Host-Prozessor 12 über das Steuerregister SR das Reset-Signal am Reset-Eingang R zurücknimmt, wird der Daten-Multiplexer 20 und der Adress-Multiplexer 22 so umgeschaltet, dass sie Daten von der Datenleitung 16 und von der Adressleitung 18 erhalten. Die Betriebsphase des SOPC-Bausteins 14 beginnt, indem über das Steuerregister SR das Reset-Signal zurückgenommen wird und die Programmdaten an einer Startadresse aus dem RAM-Baustein 24 ausgelesen werden, die dann den SOPC-Baustein 14 steuern. Damit ist es möglich, auf relativ einfache Weise und mit geringem technischen Aufwand unterschiedliche Firmware an den PLD-Baustein 10 zu übergeben.

[0018] In Fig. 2 ist ein Ausführungsbeispiel gemäß einem weiteren Erfindungsaspekt dargestellt, bei dem eine zentrale Computersteuerung 40 einen Versions-Manager zum Überwachen von Software-Modulen in verschiedenen Baugruppen eines Druckers ansteuert. Die zentrale Steuerung 40 ist über ein Bussystem 42 mit einem Hauptmodul H, weiteren Untermodulen PI, PU, PO verbunden. Das Untermodul PI ist wiederum über den Systembus 42 mit Untermodulen SM1, SM2 und SM3 verbunden, beispielsweise Schrittmotor-Module des Druckers. Die zentrale Steuerung 40 enthält einen als Software-Modul ausgebildeten Versions-Manager V. Dieser Versions-Manager V enthält Informationen über die für das System benötigten Firmware-Versionen sowie eine Beschreibungsdatei, in der angegeben ist, welche Firmware-Versionen zu welcher System-Version gehören. Beispielsweise sind diese Versionen auf einer Festplatte gespeichert, auf die der Versions-Manager V zugreift.

[0019] Jedes Software-Modul H, PI, PU, PO, SM1, SM2, SM3 enthält ein Kommunikations-Modul, welches auch als Lader bezeichnet wird. Dieses Kommunikations-Modul dient der Kommunikation des betreffenden Moduls zu seinen Nachbarmodulen und steuert den Verbindungsaufbau und das Routing der Daten. Ferner gehört zu jedem Software-Modul H bis SM3 ein Flash-ROM-Baustein (nicht dargestellt), in

welchem die aktuelle Firmware abgelegt ist. Bei einer Änderung der Firmware ist das Kommunikations-Modul dafür zuständig, dass die Firmware in diesen Flash-ROM-Baustein gebrannt wird und die Firmware gestartet wird. Weiterhin enthält das jeweilige Kommunikations-Modul Ist-Versionsdaten über das zugehörige Software-Modul und/oder die zugehörige Anwendungsumgebung, beispielsweise die Version der Baugruppe und der zugehörigen Hardware.

[0020] In Fig. 2 ist ferner ein Laptop 44 dargestellt, der über eine Schnittstelle, beispielsweise eine V 24-Schnittstelle, Zugriff auf die Software-Module H bis SM3 sowie auf die zentrale Steuerung 40 und damit auf den Versions-Manager V hat. In Fig. 2 ist diese mögliche Verbindung in Form von Strichlinien dargestellt.

[0021] Nachdem die verschiedenen Module H bis SM 3untereinander durch den Systembus 42 verbunden sind, genügt es, auf eines der dargestellten Module oder die zentrale Steuerung 40 zuzugreifen, um dann über den Systembus 42 Informationen über die weiteren Module zu erhalten oder an diese Module Daten zu übertragen. Es muss also nicht jedes Software-Modul oder jede Baugruppe mit einem Kabel vom Laptop 44 aus einzeln verbunden werden, sondern es genügt, dass eine Verbindung zu einem der Software-Module hergestellt wird, um dadurch Zugriff auf sämtliche Software-Module und die zentrale Steuerung 40 für den Drucker zu erhalten. Mit Hilfe einer solchen Verbindung ist es dann auch möglich, Firmware ausgehend von einem Software-Modul zu einem beliebigen anderen Software-Modul zu übertragen. Soll z.B. das Software-Modul SM3 mit neuer Firmware nachgeladen werden, so kann dies vom Laptop 44 aus direkt über eine V 24-Schnittstelle und die Leitung 46 erfolgen. Der Laptop 44 kann jedoch auch über die Leitungen 48, 50, 52 mit anderen Software-Modulen PI, H oder der zentralen Steuerung 40 verbunden werden, um die neue Firmware sodann über den Systembus 42 an das Software-Modul SM3 zu senden und dort die Einspeicherung in einen Flash-ROM-Baustein zu veranlassen. Im Falle, dass mehrere Software-Module die gleiche Firmware haben sollen, kann eine Übertragung dieser Firmware auch über einen Kopiervorgang erfolgen, wobei beispielsweise die Firmware des Software-Moduls SMI kopiert wird und an das Software-Modul SM3 gesendet wird.

[0022] Eine weitere Möglichkeit besteht darin, Firmware für die verschiedenen Software-Module in einen Speicher der zentralen Steuerung 40 abzulegen. Von dort aus kann dann die geforderte neue Firmware an das Software-Modul, beispielsweise das Software-Modul SM3, gesendet und dort geladen werden.

[0023] Beim Betrieb des Systems nach Fig. 2 überprüft der Versions-Manager bei bestimmten Ereignissen, beispielsweise bei einem Update der Firmware in einem der vier Module H bis SM3, ob die Ist-Versionsdaten der Software-Module mit den in seinen Verzeichnis abgespeicherten Soll-Versionsdaten für die Software-Module übereinstimmt. Falls eine Abweichung auftritt, so wird im einfachsten Fall eine Fehlermeldung erzeugt. In einer weiteren Ausgestaltung wird eine in der zentralen Steuerung 40 gespeicherte Firmware entsprechend den Soll-Versionsdaten an das betreffende Software-Modul H bis SM3 gesendet und dort vom betreffenden Kommunikations-Modul als Firmware geladen. Auf diese Weise kann das System einen ordnungsgemäßen Betriebszustand herstellen, indem automatisch der Sollzustand mit der richtigen Firmware hergestellt wird. Wenn z. B. bei einem Ersatzteilaustausch in einer Baugruppe, die zu einem Software-Modul H bis SM3 gehört, eine andere Firmware-Version geladen wird, als sie gemäß der Soll-Versionsdaten im Versions-Manager gespeichert sind, so kann dieser Fehler im System schnell erkannt werden und gegebenenfalls behoben werden. Auf diese Weise wird die Betriebssicherheit erhöht und der Wartungsaufwand und der Service verringert. Nachdem von einem beliebigen Software-Modul bzw. der zugehörigen Baugruppe aus auf die anderen Software-Module und Baugruppen durch einen externen Rechner zugegriffen werden kann, ist das dargestellte System einfach zu warten und es kann eine flexible Überprüfung des ordnungsgemäßen Betriebs erfolgen.

[0024] Im folgenden wird ein Ausführungsbeispiel gemäß einem weiteren Erfindungsaspekt beschrieben, bei dem durch eine Versionskontrolle sichergestellt wird, dass nur zusammen funktionsfähige Software-Module in einem Drucker zum Einsatz gelangen, bzw. dass bei nicht vorhandener.Funktionsfähigkeit der verschiedenen Software-Module eine Fehlermeldung erzeugt wird. Beim vorliegenden Beispiel wird eine schnittstellenbasierte Online-Versionsüberprüfung durchgeführt. Dieser Erfindungsaspekt kann bei einem Druckersystem angewendet werden, das mehrere Software-Module umfasst. Es ist jedoch auch außerhalb von Druckeranwendungen allgemein anwendbar.

[0025] Ein mehrere Software-Module umfassendes System kommuniziert untereinander auf der Basis eines Netzwerk-Protokolls. Beispielsweise kommt hierfür das bekannte SNMP-Kommunikationsprinzip (Simple Network Management Protocol) in Betracht, welches ein Standard-Protokoll für das Netzmanagement in der Internet-Welt zur Kommunikation von Netzwerkelementen untereinander ist. Bei einem solchen System hat jedes Software-Modul einen Software-Agenten, d.h. eine für das Software-Modul zugehörige Teilsoftware, die für die Verwaltung (Konfiguration, Parameter, Funktionsbeschreibung etc.) dieses Software-Modul und die Kommunikation mit anderen Software-Modulen zu verwaltungstechnischen Zwecken zuständig ist.

[0026] Fig. 3 zeigt schematisch die Verknüpfung von Verweisen bei einer Versionsprüfung. Zu Softwaremodulen AA, BB und CC sind die Software-Agenten aa, bb und cc dargestellt. Jeder Software-Agent enthält Ist-Versionsdaten, z.B. „aaAgent-

Version 01.01.13", und eine Versionsnummer der Schnittstelle (Mib), beispielsweise „aaMibVersion 01.01.24" (Mib: Management information base). Diese Mib-Daten bezeichnen Datenbestände, mit deren Hilfe das Netzmanagement System SNMP alle zu verwaltenden Objekte (z.B. Geräte, Rechner, Server, Ruder etc.) im Netz verwaltet. Die Versionsdaten sind als Gruppen von zweistelligen Dezimalzahlen dargestellt. Diese Nummerierung hat den Vorteil, dass eine Aufwärtskompatibilität und eine Abwärtskompatibilität leicht anhand von Ungleichungen den logischen Operatoren „≥" oder „≤" durchgeführt werden kann. Für jeden Agenten aa, bb, cc nach Fig. 3 sind als Soll-Versionsdaten Datenblöcke 60, 62, 64 vorgesehen, in welchen OID-Elemente auf die entsprechenden Datenfelder der Software-Agenten verweisen. Diese OID-Elemente (object identifier) sind eindeutige Adressen für Objekte innerhalb der weltweit vorhandenen Mib-Struktur, wobei jeder Knoten der Mib-Struktur eine Nummer innerhalb der kompletten Adresse (OID) darstellt. Der Verweis auf die Soll-Versionsnummer ist somit flexibel, da der betreffende Software-Agent und Speicherort durch das OID-Element als Verweis angesprochen wird. Als Werte zum OID-Element sind jeweils Gruppen mit zweistelligen Dezimalzahlen getrennt durch Punkte angegeben.

[0027] In Fig. 3 ist für die drei Software-Agenten aa, bb und cc durch Pfeile angedeutet, wie die Versionsprüfung durchgeführt wird. Diese Versionsprüfung erfolgt dezentral für jeden Software-Rgenten.

[0028] Beispielsweise benötigt der Software-Agent aa für den ordnungsgemäßen Betrieb des i.hm zugeordneten Software-Moduls die im Block 60 angegebenen Soll-Versionsdaten anderer Software-Module. Demgemäss bedeutet Pfeil 66, dass die eingetragene OID auf den Eintrag "bbAgentVersion" im Software-Agenten bb verweist und überprüft wird, ob die Soll-Versionsdaten „bbAgentVersion" mit Wert „01.01.12" mit den Ist-Versionsdaten im Software-Agenten bb „bbAgentVersion" mit Wert „01.01.13" übereinstimmt oder nicht. Im vorliegenden Falle ist die Übereinstimmung in der letzten Dezimalgruppe nicht gegeben, welche ein Maß für den Kompatibilitätsgrad ist. Gemäß einer Übereinkunft bedeutet eine Änderung in der letzten Ziffer, dass kleinere Änderungen vorgenommen wurden und diese Version aufwärtskompatibel ist. In diesem Fall wird keine Fehlermeldung erzeugt, weil eine neuere Version vorhanden ist als mindestens benötigt wird. In einem Falle mit fehlender Kompatibilität wird eine Fehlermeldung erzeugt. Auf gleiche Weise überprüft der Software-Agent aa die notwendigen Soll-Versionsdaten gemäß den Verweispfeilen 68 und 70. Auf ähnliche Weise erfolgt die Versionsüberprüfung für den Software-Agenten bb und den Software-Agenten cc gemäß den eingezeichneten Verweispfeilen.

[0029] Die dezentrale Überprüfung der vorhandenen Versionen der verschiedenen Software-Module durch die Software-Agenten erfolgt beispielsweise nach jedem Einschalten des Systems, nach vorbestimmten Zeitabständen oder nach einer Änderung der Software.

[0030] Obgleich in den Zeichnungen und in der vorhergehenden Beschreibung bevorzugte Ausfizhrungsbeispiele aufgezeigt und detailliert beschrieben sind, sollte dies als rein beispielhaft und die Erfindung nicht einschränkend angesehen werden. Es wird darauf hingewiesen, dass nur die bevorzugten Ausführungsbeispiele dargestellt und beschrieben sind und sämtliche Veränderungen und Modifizierungen, die derzeit und künftig im Schutzumfang der Erfindung liegen, geschützt werden sollen.

Bezugszeichenliste

10	PLD-Baustein
12	Host-Prozessor
14	SOPC-Baustein
16	Datenleitungen
18	Adressleitungen
20	Daten-Multiplexer
22	Adress-Multiplexer
24	RAM-Baustein
26	Registerschnittstelle
SR	Steuer-Register
DR	Daten-Register
STR	Startadressen-Register
28	Adresszähler
30	Flash-ROM-Speicher
32,34	Datenleitungen
40	Zentrale Steuerung
M	Versions-Manager
42	Bus-System
H	Hauptmodul
PI,PU,PO	Untermodule
SM1,SM2,SM3	Untermodule
44	Laptop
46	Leitung
48,50,52	Leitungen
AA,BB,CC	Software-Module
aa,bb, cc	Software-Agenten
60	Datenblock
66, 68, 70	Verweispfeile

Patentansprüche

1. Einrichtung zum Betreiben eines PLD-Bausteins,
bei der der PLD-Baustein (10) einen Microcontroller 14, der auf einen RAM-Baustein (24) zugreift, einen Daten-Multiplexer (20) zum Zugriff auf Daten im RRM-Baustein (24) sowie einen Adress-Multiplexer (22) zum Zugriff auf Speicheradressen im RAM-Baustein (24) enthält,
der PLD-Baustein (10) mit einem Host-Prozessor (12) verbindbar ist,
in einer Ladephase der Host-Prozessor (12) Programmdaten zum Steuern des Microcontrollers (14) über den Daten-Multiplexer (20) und gesteuert durch den Adress-Multiplexer (22) in den RAM-Baustein

(24) lädt,
und bei der in einer Betriebsphase der Microcontroller (14) auf die Programmdaten im RAM-Baustein (24) unter Einschaltung des Daten-Multiplexers (20) und des Adress-Multiplexers (22) zugreift.

2. Einrichtung nach Anspruch 1, bei der der Microcontroller (14) ein SOPC-Baustein ist, der einen Reset-Eingang (R) und weiterhin mit dem Adress-Multiplexer (22) verbundene Adressleitungen (18) und mit dem Daten-Multiplexer (20) verbundene Datenleitungen (16) hat.

3. Einrichtung nach Anspruch 2, bei der in der Ladephase der Host-Prozessor (12) den Microcontroller (14) in den Reset-Zustand schaltet, die Datenleitungen des Daten-Multiplexers (20) mit Datenleitungen des Host-Prozessors (12) und der Adress-Multiplexer (22) mit einem steuerbaren Adresszähler (28) verbunden ist.

4. Einrichtung nach einem der vorhergehenden Ansprüche, bei der der Host-Controller (12) mit einem ROM-Baustein (30), vorzugsweise einem Flash-ROM-Baustein verbunden ist, der die in den RAM-Baustein (24) zu ladenden Programmdaten enthält.

5. Einrichtung nach einem der vorhergehenden Ansprüche, bei der der Host-Prozessor (12) ferner mit einem Bus-System für ein Computer-Netzwerk verbunden ist.

6. Einrichtung nach einem der vorhergehenden Ansprüche, bei der der PLD-Baustein (10) eine Register-Schnittstelle (26) enthält, auf die der Host-Prozessor (12) zugreift.

7. Einrichtung nach Anspruch 6, bei der die Register-Schnittstelle ein Steuer-Register (SR), ein Daten-Register (DR) und ein Startadressen-Register (STR) hat.

8. Einrichtung nach Anspruch 7, bei der über das Steuer-Register (SR) auf den Reset-Eingang (R) des SOPC-Bausteins (14) zugegriffen wird.

9. Einrichtung nach Anspruch 8, bei dem über das Startadressen-Register (STR) auf den Adress-Zähler (28) zugegriffen wird und eine Startadresse für den RAM-Baustein (24) übergeben wird.

10. Verfahren zum Betreiben eines PLD-Bausteins,
bei dem der PLD-Baustein (10) einen Microcontroller 14, der auf einen RAM-Baustein (24) zugreift, einen Daten-Multiplexer (20) zum Zugriff auf Daten im RAM-Baustein (24) sowie einen Adress-Multiplexer (22) zum Zugriff auf Speicheradressen im RAM-Baustein (24) enthält,
der PLD-Baustein (10) mit einem Host-Prozessor (12) verbindbar ist,
in einer Ladephase der Host-Prozessor (12) Programmdaten zum Steuern des Microcontrollers (14) über den Daten-Multiplexer (20) und gesteuert durch den Adress-Multiplexer (22) in den RAM-Baustein (24) lädt,
und bei dem in einer Betriebsphase der Microcontroller (14) auf die Programmdaten im RAM-Baustein (24) unter Einschaltung des Daten-Multiplexers (20) und des Adress-Multiplexers (22) zugreift.

11. Verfahren zum Betreiben eines Datenverarbeitungs-Systems,
bei dem das System eine zentrale Steuerung (40) umfasst, die einen als Software-Modul ausgebildeten Versions-Manager (V) enthält,
die zentrale Steuerung (40) über ein Bus-System (42) mit mehreren Software-Modulen (H, PI, PU, PO, SM1, SM2, SM3) verbunden ist, die jeweils ein Kommunikations-Modul enthalten, welches die Kommunikation der Software-Module untereinander und den Start der für das jeweilige Software-Modul zuständige Firmware steuert,
das jeweilige Kommunikations-Modul Ist-Versionsdaten über das zugehörige Software-Modul und/oder die zugehörige Anwendungsumgebung enthält,
der Versions-Manager ein Verzeichnis der Soll-Versionsdaten der verschiedenen Software-Module umfasst,
bei einer Änderung des Systems der Versions-Manager auf die Ist-Versionsdaten der verschiedenen Software-Module zugreift und diese mit den Soll-Versionsdaten vergleicht,
und bei dem bei einer Abweichung eine Fehlermeldung erzeugt wird.

12. Verfahren nach Anspruch 11, bei dem bei einer Abweichung der Ist-Versionsdaten von den Soll-Versionsdaten eine in der zentralen Steuerung (40) gespeicherte Firmware entsprechend den Soll-Versionsdaten an das betreffende Software-Modul gesendet und vom betreffenden Kommunikations-Modul als Firmware geladen wird.

13. Verfahren nach Anspruch 11 oder 12, bei dem die Firmware in einem Flash-ROM-Baustein enthalten ist, welche vom zugehörigen Kommunikations-Modul geladen wird.

14. Verfahren nach einem der vorhergehenden Ansprüche, bei dem mehrere oder alle Software-Module eine Schnittstelle zu einem externen Computer (44) haben, wobei über diese Schnittstelle Zugriff auf das Kommunikations-Modul oder über das Bus-System (42) auf andere Kommunikations-Module erfolgen kann.

15. Verfahren nach einem der vorhergehenden Ansprüche, bei dem über einen externen Rechner

(42) auf den Versions-Manager zugegriffen wird und Firmware in die zentrale Steuerung (40) gespeichert wird.

16. Einrichtung zum Betreiben eines Datenverarbeitungs-Systems,
bei der das System eine zentrale Steuerung (40) umfasst, die einen als Software-Modul ausgebildeten Versions-Manager (V) enthält,
die zentrale Steuerung (40) über ein Bus-System (42) mit mehreren Software-Modulen (H, PI, PU, PO, SM1, SM2, SM3) verbunden ist, die jeweils ein Kommunikations-Modul enthalten, welches die Kommunikation der Software-Module untereinander und den Start der für das jeweilige Software-Modul zuständige Firmware steuert,
das jeweilige Kommunikations-Modul Ist-Versionsdaten über das zugehörige Software-Modul und/oder die zugehörige Anwendungsumgebung enthält,
der Versions-Manager ein Verzeichnis der Soll-Versionsdaten der verschiedenen Software-Module umfasst,
bei einer Änderung des Systems der Versions-Manager auf die Ist-Versionsdaten der verschiedenen Software-Module zugreift und diese mit den Soll-Versionsdaten vergleicht,
und bei der bei einer Abweichung eine Fehlermeldung erzeugt wird.

17. Verfahren zum Betreiben eines Datenverarbeitungs-Systems,
bei dem das System mehrere Software-Module umfasst, die untereinander auf der Basis eines Netzwerk-Protokolls kommunizieren,
jedes Software-Modul einen Software-Agenten enthält, der Ist-Versionsdaten über das eigene Software-Modul und Soll-Versionsdaten über die für seine Funktion benötigten weiteren Software-Module enthält,
der jeweilige Software-Agent auf die Versionsdaten der Software-Agenten der benötigten Software-Module zugreift und sie erfaßt,
und bei dem in einer Versionsprüfung diese Soll-Versionsdaten der benötigten Software-Module und die erfassten Ist-Versionsdaten der benötigten Software-Module auf Kompatibilität überprüft werden.

18. Verfahren nach Anspruch 17, bei dem als Netzwerk-Protokoll das SNMP (Simple Network Management Protokoll) verwendet wird.

19. Verfahren nach einem der vorhergehenden Ansprüche, bei dem die Versionsdaten als Gruppen von zweistelligen Dezimalzahlen dargestellt werden.

20. Verfahren nach Anspruch 19, bei dem eine Gruppe von Dezimalzahlen die Aufwärtskompatibilität und Abwärtskompatibilität kennzeichnet und wobei bei der Versionsprüfung ein Vergleich auf der Basis einer Ungleichung erfolgt.

21. Verfahren nach einem der vorhergehenden Ansprüche, bei dem bei der Versionsprüfung im Falle einer Inkompatibilität ein Eintrag in einer zentralen Fehlertabelle erfolgt.

22. Verfahren nach einem der vorhergehenden Ansprüche, bei dem die Versionsprüfung bei jedem Einschalten des Systems erfolgt.

23. Verfahren nach einem der vorhergehenden Ansprüche, bei dem das System in einem Hochleistungsdrucker enthalten ist.

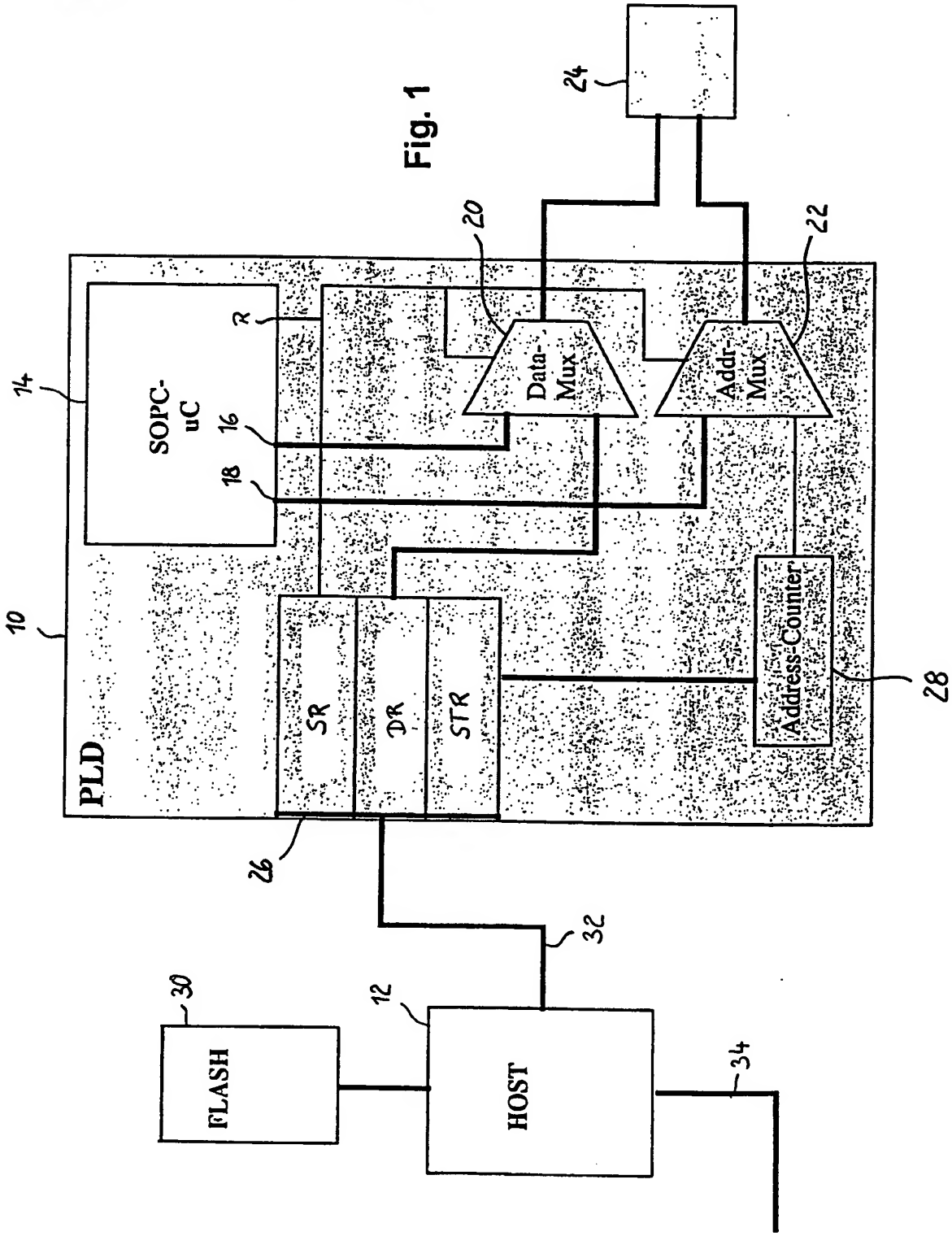
24. Verfahren nach einem der vorhergehenden Ansprüche 17 bis 23, bei dem die Software-Agenten (aa, bb, cc) die Daten bereitstellen und ein ausgewählter Software-Agent zentral die Überprüfung durchführt.

25. Einrichtung zum Betreiben eines Datenverarbeitungs-Systems,
bei der das System mehrere Software-Module umfasst, die untereinander auf der Basis eines Netzwerk-Protokolls kommunizieren,
jedes Software-Modul einen Software-Agenten enthält, der Ist-Versionsdaten über das eigene Software-Modul und Soll-Versionsdaten über die für seine Funktion benötigten weiteren Software-Module enthält,
der jeweilige Software-Agent auf die Versionsdaten der Software-Agenten der benötigten Software-Module zugreift und sie erfaßt,
und bei der in einer Versionsprüfung diese Soll-Versionsdaten der benötigten Software-Module und die erfassten Ist-Versionsdaten der benötigten Software-Module auf Kompatibilität überprüft werden.

Es folgen 3 Blatt Zeichnungen

Anhängende Zeichnungen

Fig. 1



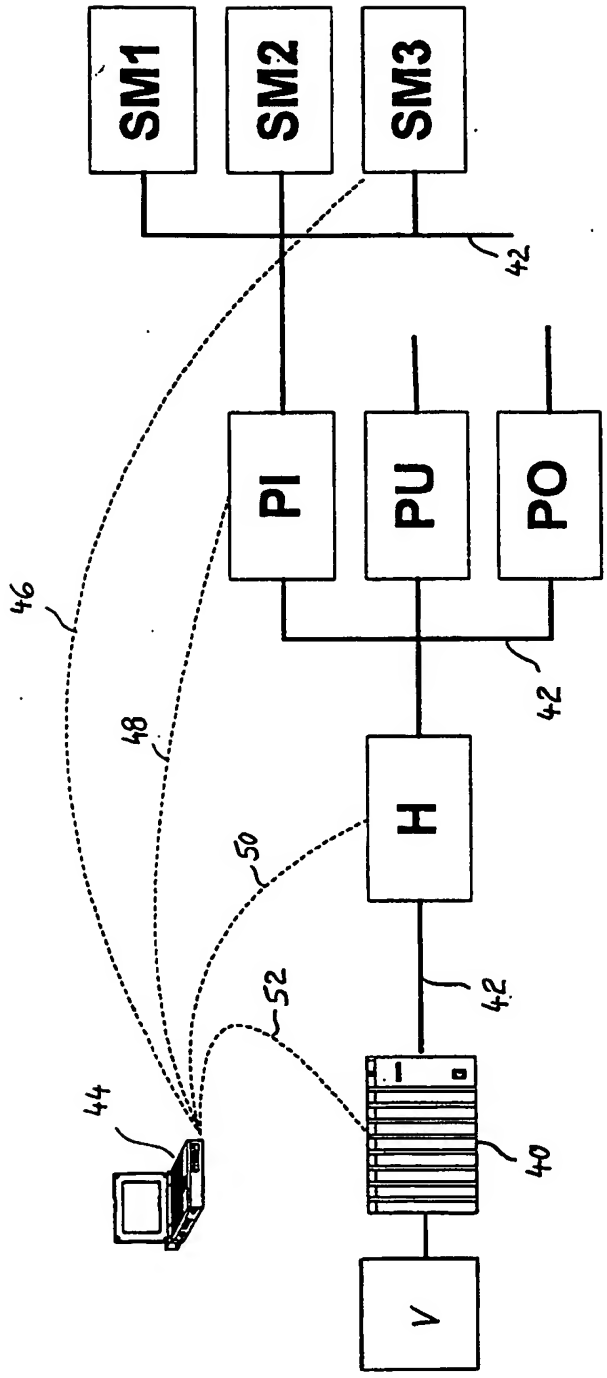


Fig. 2

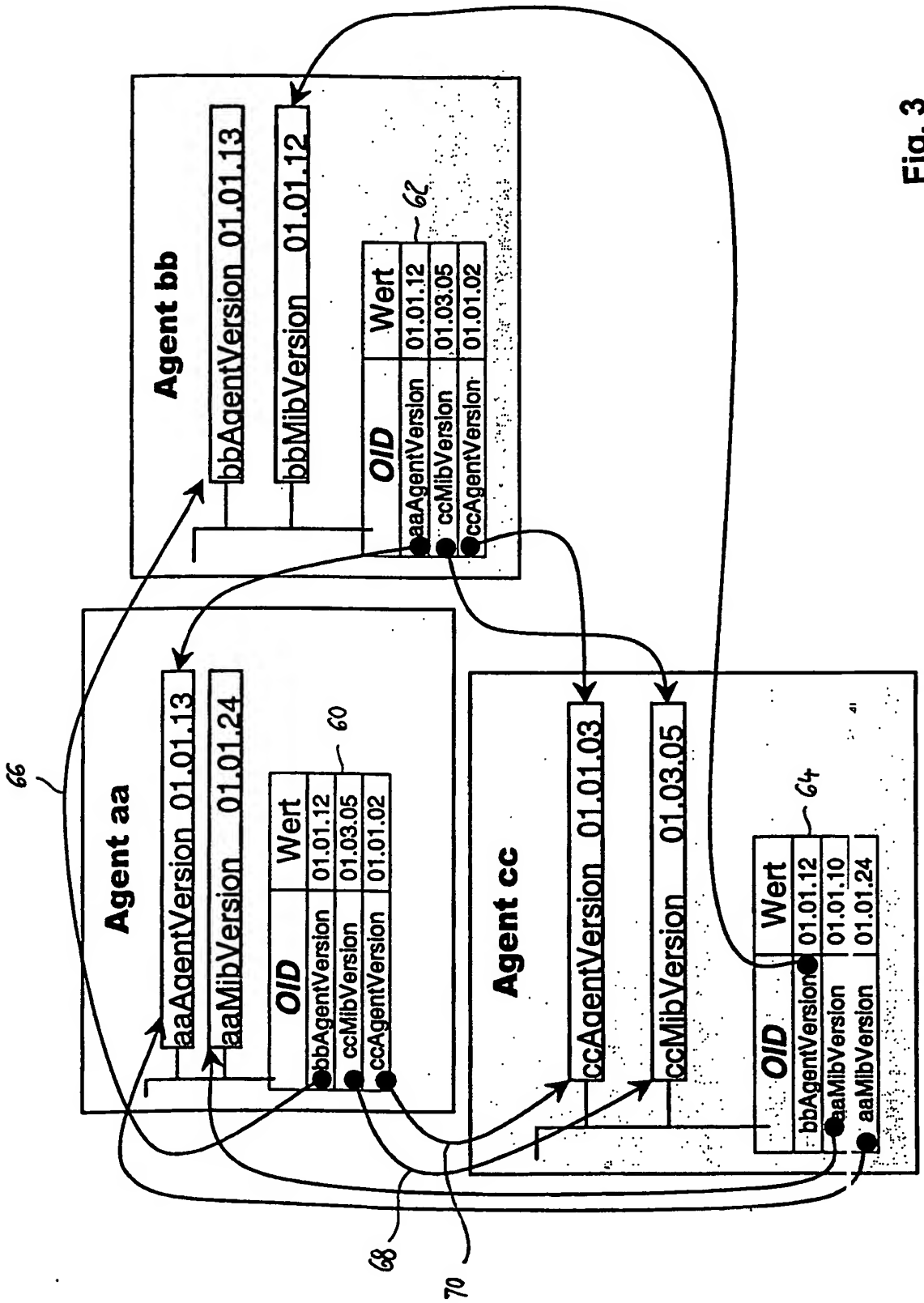


Fig. 3